PerTurbo manifold learning algorithm for weakly labelled hyperspectral image classification

Laëtitia Chapel, Thomas Burger, Nicolas Courty, Sébastien Lefèvre

PerTurbo manifold learning algorithm for weakly labelled hyperspectral image classification

Laetitia Chapel, Thomas Burger, Nicolas Courty, Sébastien Lefèvre

***Abstract*—Hyperspectral data analysis has been given a growing attention due to the scientific challenges it raises and the wide set of applications which can benefit from it. Classification of hyperspectral images has been identified as one of the hottest topics in this context, and has been mainly addressed by discriminative methods such as SVM. In this paper, we argue that generative methods, and especially those based on manifold representation of classes in the hyperspectral space, are relevant alternatives to SVM. To illustrate our point, we focus on the recently published PerTurbo algorithm and benchmark against SVM this generative manifold learning algorithm in the context of hyperspectral image classification. This choice is motivated by the fact that PerTurbo is fitted with numerous interesting properties, such as (1) low sensitivity to dimensionality curse, (2) high accuracy in weakly labelled images classification context (few training samples), (3) straightforward extension to on-line setting, (4) interpretability for the practitioner. The promising results call for an up-to-date interest towards generative algorithms for hyperspectral image classification.**

***Index Terms*—Classification, remote sensing, hyperspectral images, PerTurbo algorithm, generative method, manifold learning, Support Vector Machines, low-sized training sets.**

I. Introduction

A. Context

The classification of hyperspectral images had been a subject of interest for the remote sensing community for the last decade, due to the generalisation of hyperspectral sensors and their recent advances. Hyperspectral data are composed of hundreds of images corresponding to different spectral bands. Classification of such images is still a challenging task as indicated in a recent survey [1]. Indeed, it entails processing a potential huge amount of data that are high dimensional, leading to significant time and memory requirements. From a methodological viewpoint, many classifiers are not appropriate in this context as they suffer from the dimensionality curse: the classification accuracy decreases with the dimension of the data when the number of available pixels for training is fixed. Moreover, training samples are usually limited, costly and quite difficult to obtain in remote sensing scenarios, which makes the Hughes phenomenon even more critical [2]. As an example, we can cite the under-achievements of Gaussian classifiers or neural networks techniques [3], [4]. More recently, Support Vector Machines (SVM) [5] have received particular attention in the context of hyperspectral image classification [6] as they alleviate the dimensionality curse (as most of the methods based on the kernel trick). Since then, some adaptations have been developed, *e.g.* kernel functions that take into account the spatial neighborhood information [7], [8].

B. Motivations

Although hyperspectral data live in a high dimensional space, the spectral correlation between bands is high and it is very unlikely that they occupy the whole space in an anisotropic manner: a manifold assumption then makes sense. Manifold learning algorithms assume that the original high dimensional data actually lie on an embedded lower dimensional manifold. Similarly to numerous other image processing or computer vision classification problems, one expects the data to live in a Riemannian [9] or statistical [10] manifold, the geometric understanding of which will help the interpretation of the classification model (decomposition of a class into several sub-classes, computation of the distance between each class, etc.). Thus, from a practitioner point of view, it appears sensible to have a classifier built on this manifold assumption. Techniques like PCA or Minimum Noise Fraction [11] can be applied to hyperspectral images in order to determine their intrinsic dimension. The mapping of the data from high to low dimensional spaces can also be learnt thanks to learning algorithms such as ISOMAP [12] or local linear embedding [13]. Their applications to hyperspectral image data have been proposed recently [14], [15], [16], [17], [18] and the results indicate that they can be efficiently represented and characterised in low dimensional spaces. Indeed, it has been identified in [1] as a way to deal with the geometry of hyperspectral data (complex and dominated by nonlinear structures) and thus to further improve the classification accuracy.

In our context, another path of interest to improve the classification performances is to focus on generative algorithms, that naturally entail a description of each class. Of course, a cornerstone of the statistical learning theory [5] is that learning a boundary between classes (as it is done with discriminative classifiers) is a simpler problem than training a model for each of them. As a direct consequence, one expects a classifier based on generative models to be less efficient than an optimal margin classifier such as SVM. However, from a theoretical point of view, generative learning is often more efficient than discriminative learning when the number of features is large compared to the number of training samples [19] while discriminative models are often better asymptotically. One reason is that the latter tends to overfit when the number of training examples is low (more practically, achieving an appropriate regularisation or tuning of SVM becomes difficult with few data). Ideal classifier is therefore likely to change when the number of training samples increases, and some attempts have been done to determine the threshold above which the classifiers should be switched in an on-line setting [20]. In the context of hyperspectral image classification, generative classifiers are particularly interesting since, as pointed out in [1], "supervised classification faces challenges related with the unbalance between high dimensionality and limited availability of training samples". Nevertheless, comparison of the two models is a perennial topic as the superiority of generative classifiers may not be systematically observed, depending on the data and model specification [21]. Indeed, the first attempts in this direction in the remote sensing community were not successful so far, as illustrated in [22] where SVM still outperform kernel Fisher discriminant analysis when the training set size is low.

In addition, generative classifiers have appealing properties that make them adapted to interactive image data mining. For instance, when a class is split in two or when a hierarchical classification is sought, only a part of the models needs to be re-learnt; only the new class that could be added to the initial set of classes needs a specific training, the other models remaining unaltered (this last property can

L. Chapel, N. Courty and S. Lefèvre are with Univ. Bretagne-Sud, UMR 6074, IRISA, 56000 Vannes, France. e-mail: name.surname@univ-ubs.fr.

T. Burger is with Université Grenoble-Alpes, CEA (iRSTV/BGE), INSERM (U1038), CNRS (FR3425), 38000 Grenoble, France. e-mail: name.surname@cea.fr.

rarely be found in discriminative methods, see for instance [23]). For all these reasons, we believe that the use of generative models for hyperspectral image classification should be further investigated.

C. Contributions

In this article, we focus on the recently published PerTurbo algorithm [24] for hyperspectral image classification. The rationale for testing PerTurbo in this context is due to its following appealing properties: it belongs to the family of generative classifiers (and hence we may expect that it performs well with low training set sizes) and is based on manifold assumptions (and then performs well in high dimensional spaces). Description of the algorithm is given in section II: the main idea behind it is to describe each class by an intrinsic representation based on a Laplace-Beltrami operator approximation [25] that can be interpreted in terms of kernel space. This characterisation allows the association of a topologic piece of information to each class that describes its spatial distribution in the input space. This geometric characterisation is computed for each class, before and after the addition of the test sample that is to be classified. The extent to which the geometric characterisation is perturbed by the test sample is a good evidence on its similarity to the class. Thus, this sample is classified into the class that was the least perturbed by its addition. The geometric description can also be used to describe classes and to derive some additional information, *e.g.* measure of how classes are well-separated. We also address a particular challenge raised in [1], *i.e.* computational complexity of hyperspectral data classification. To do so, we discuss the algorithm complexity, as well as a step towards a fast version of the algorithm. In section III, we benchmark the algorithm against SVM in the low training set size setting and show how the geometric characterisation gives an insight into the separability of classes. Finally, we draw some conclusions and give some perspectives.

This article extends a preliminary work which was presented at IGARSS conference [26]. The contributions of this paper are the following: i) we give some theoretical arguments about the situations in which PerTurbo could be an interesting alternative to state-of-the-art discriminative algorithms like SVM; ii) we report the results of rigourous experiments that show that, as stated by the theory, PerTurbo outperforms SVM in weakly labelled image classification context; iii) we discuss a fast version of the algorithm in order to mitigate the limitations due to its high computational complexity; iv) we provide a similarity measure between pairs of classes based on their geometric characterisation that allows one to obtain beforehand an insight into the separability of the classes, and hence into the expected performances of the classification algorithm.

II. PerTurbo

A. Manifold class description and classification algorithm

PerTurbo was initially introduced in a pure machine learning context. Here, we simply summarise its most important features, and the interested reader should refer to [24].

We consider a classical machine learning problem, where one seeks for a function that best labels a set of unlabelled examples. We denote

$$\mathcal{S} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_N, y_N)\} \in \mathbb{R}^d \times \{u_1, ..., u_L\} \quad (1)$$

the training set of N training samples $\boldsymbol{x}_i$ of label y_i, and $\mathcal{S}_\ell \subset \mathcal{S}$ is the set of all the N_ℓ training examples with label u_ℓ. The idea behind PerTurbo is to build the predictive function in an implicit manner, within two steps. In the first step, the geometry of the set of training examples for each class $\mathcal{S}_\ell$ is characterised. Then, a similarity metric adapted to these sets of geometric models is derived, so that the predictive function reads as the minimum distance of a test sample $\boldsymbol{\eta}$ to those models.

More formally, each set $\mathcal{S}_\ell$ is assumed to be embedded in a dedicated Riemannian manifold $\mathcal{M}_\ell$, whose geometric structure can be expressed in terms of the Laplace-Beltrami operator. As the manifolds corresponding to the classes are unknown (only the training examples they embed are accessible), it is generally not possible to find an analytical expression of this operator. However, it has been established in [27] that this operator can be efficiently approximated by the heat kernel (modeling the propagation of a variation of temperature along a manifold), this latter being in turn approximated by the spectrum of the Gaussian kernel $K(\mathcal{S}_\ell)$, the Gram matrix of the training set $\mathcal{S}_\ell$, whose (ith, jth) term is given by:

$$K_{ij}(\mathcal{S}_\ell) = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^T \cdot \phi(\boldsymbol{x}_j) = \exp\left(-\gamma \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2\right), \quad (2)$$

where $k(\cdot,\cdot)$ is a kernel function, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j \in \mathcal{S}_\ell$, $\cdot^T$ being the transpose of a matrix or a vector, ϕ is the mapping from the original space into the Hilbert space reproducing the Gaussian kernel (also called feature space, Reproducing Kernel Hilbert Space or RKHS), $\|\cdot\|$ is the Euclidean norm and $\gamma \in \mathbb{R}_+^*$ accounts for the variance of the Gaussian kernel.

When a test sample $\boldsymbol{\eta} \in \mathbb{R}^d$ is considered, a similarity measure between a class and $\boldsymbol{\eta}$ can be derived from the extent to which its inclusion to the class changes the geometric characterisation of the associated manifold $\mathcal{M}_\ell$. More formally, the projection of sample $\boldsymbol{\eta}$ onto the subspace spanned by $\phi(\mathcal{S}_\ell)$ is given by:

$$r(\boldsymbol{\eta} \to \mathcal{M}_\ell) = K(\mathcal{S}_\ell)^{-1/2} \cdot k(\mathcal{S}_\ell, \boldsymbol{\eta}) \quad (3)$$

where the ith term of $k(\mathcal{S}_\ell, \boldsymbol{\eta})$ is $k(\boldsymbol{x}_i, \boldsymbol{\eta})$, with $\boldsymbol{x}_i \in \mathcal{S}_\ell$. The *perturbation measure* of class u_ℓ by sample $\boldsymbol{\eta}$ then reads:

$$\begin{aligned}\tau(\boldsymbol{\eta}, \mathcal{M}_\ell) &= \|\phi(\boldsymbol{\eta})\|^2 - \|r(\boldsymbol{\eta} \to \mathcal{M}_\ell)\|^2 \\ &= 1 - k(\mathcal{S}_\ell, \boldsymbol{\eta})^T \cdot K(\mathcal{S}_\ell)^{-1} \cdot k(\mathcal{S}_\ell, \boldsymbol{\eta}),\end{aligned} \quad (4)$$

and it quantifies the perturbation of the manifold $\mathcal{M}_\ell$ when $\boldsymbol{\eta}$ is added to class u_ℓ. Test sample $\boldsymbol{\eta}$ is then classified into the class that provides the smallest perturbation, *i.e.*

$$\arg\min_\ell \tau(\boldsymbol{\eta}, \mathcal{M}_\ell). \quad (5)$$

Thus, in order to account for non-linear data processing, PerTurbo can be seen either as a subspace classifier where vector subspaces are replaced by Riemannian manifolds, or as a classical subspace classifier which operates in a kernelised space. As such, the ideas underlying PerTurbo are somehow related to that of a host of other classification methods based on subspace projections, such as, for instance, [28]. Interestingly enough, these two complementary interpretations justify the appealing properties of PerTurbo: as a generative model based on the manifold geometry of the datasets, it helps interpretability of the models, while, as a classifier working in the feature space, distances in the high dimensional input space are replaced by similarity measures independent of the input space. This makes the algorithm less sensitive to the dimensionality curse, a necessary property for hyperspectral image classification.

Naturally, PerTurbo works as long as the perturbation measure $\tau(\boldsymbol{\eta}, \mathcal{M}_\ell)$ is defined, hence as long as $K(\mathcal{S}_\ell)$ is invertible $\forall \ell \leq L$, which is the case since it is a symmetric positive definite matrix. However, for numerical analysis issues, $K(\mathcal{S}_\ell)$ might not be invertible. Yet, it is always possible to compute its pseudo-inverse or to consider regularisation techniques that find a close invertible matrix. In addition, the regularisation of the models can also be used in order to avoid overfitting and high sensitivity to features noise. In this paper, we consider the case of Tikhonov regularisation where a regularised version of $K(\mathcal{S}_\ell)$ is used:

$$\widetilde{K}(\mathcal{S}_\ell) = K(\mathcal{S}_\ell) + \lambda \cdot \mathbf{I}_{N_\ell} \quad (6)$$

where $\lambda > 0$ is the Tikhonov factor. The main interest of such a regularisation is that it makes the Gram matrix (the spectrum of which is equivalent to that of the covariance matrix class u_ℓ) less sensitive to outliers. Hence, even in the case where $K(\mathcal{S}_\ell)$ is invertible, it is possible to boost performances by tuning λ to a value which is adapted to the inner-class covariance matrices of the dataset.

In addition, there is another way to improve the general frame of PerTurbo in the presence of noisy spectral channels. In a PCA-like

way, we can perform an implicit regularisation via the reduction of the noise classically associated to the less expressive dimensions. To do so, the spectrum of the Gram matrix can be truncated, so that its smallest eigenvalues below a given threshold are gotten rid of. The regularisation parameter λ is then the percentage of the sum of the total eigenvalues kept. Mainly because it relates PerTurbo to the well-known Graph Laplacian Eigenmaps framework [29], this regularisation is appealing from a theoretical point of view. Those two regularised versions of PerTurbo imply the setting of one additional parameter λ, related to the magnitude of the regularisation.

B. Interpretability of the model: similarity between classes

The geometric characterisation of each class by the kernel matrix $K(\mathcal{S}_\ell)$ carries extra information that can be used to describe classes. Here, we illustrate the advantage of having such a representation by deriving a measure of similarity between pairs of class models that allows one to give beforehand an insight into the separability of the classes, and hence into the expected performances of the classification algorithm that can be used on the data.

The projection of sample $\boldsymbol{\eta}$ on manifold $\mathcal{M}_\ell$ defined in Eq. (3) can be generalised in order to project all the samples $\mathcal{S}_{\ell_1}$ of a given class u_{ℓ_1} into the subspace spanned by $\phi(\mathcal{S}_{\ell_2})$:

$$r(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2}) = K(\mathcal{S}_{\ell_2})^{-1/2} \cdot k(\mathcal{S}_{\ell_2}, \mathcal{S}_{\ell_1}) \tag{7}$$

Hence, the Gram matrix associated to this projection is

$$\begin{aligned} K(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2}) &= r(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2})^T \cdot r(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2}) \\ &= k(\mathcal{S}_{\ell_2}, \mathcal{S}_{\ell_1})^T \cdot K(\mathcal{S}_{\ell_2})^{-1} \cdot k(\mathcal{S}_{\ell_2}, \mathcal{S}_{\ell_1}). \end{aligned} \tag{8}$$

$K(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2})$ is a surrogate kernel of $K(\mathcal{S}_{\ell_1})$ [30]: both kernels are constructed on the same set of eigenvectors and eigenvalues, but they are evaluated on different sets. Its definition can then be used to compare Gram matrices $K(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2})$ and $K(\mathcal{S}_{\ell_1})$ as they are both constructed on the same set of points: classes with similar geometry will have "similar" kernel matrices while those with manifolds lying in different spaces will have "different" kernel matrices. The similarity between pairs of kernel matrices is hence an indicator of the performances of the classification method and is a proxy of how classes are well-separated. The degree of agreement between two matrices can be evaluated thanks to the empirical alignment of the kernel matrix $K(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2})$ with the kernel matrix $K(\mathcal{S}_{\ell_1})$ [31] with respect to the sample $\mathcal{S}_{\ell_1}$:

$$A(\mathcal{S}_{\ell_1}, K(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2}), K(\mathcal{S}_{\ell_1})) = \frac{\langle K(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2}), K(\mathcal{S}_{\ell_1}) \rangle_F}{\sqrt{\langle K(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2}), K(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2}) \rangle_F \langle K(\mathcal{S}_{\ell_1}), K(\mathcal{S}_{\ell_1}) \rangle_F}} \tag{9}$$

where $\langle C, D \rangle_F$ stands for the inner product between matrices C and D, *i.e.* $\langle C, D \rangle_F = \sum_{i,j} C_{ij} D_{ij}$.

C. Computational complexity: towards a fast version of PerTurbo

First, we note that the algorithm can be straightforwardly extended to the on-line setting. When a new sample $(\boldsymbol{x}_{N+1}, y_{N+1})$ is added to training set $\mathcal{S}_\ell$, the inverse of the updated Gram matrix $K(\mathcal{S}_\ell \cup \boldsymbol{x}_{N+1})$ can easily be computed iteratively, using the Schur complement [32], which is here equal to $1/\tau(\boldsymbol{x}_{N+1}, \mathcal{M}_\ell)$, denoted τ^{-1} in the following equation:

$$\begin{aligned} K(\mathcal{S}_\ell \cup \boldsymbol{x}_{N+1})^{-1} &= \begin{bmatrix} K(\mathcal{S}_\ell) & k(\mathcal{S}_\ell, \mathbf{x}_{N+1}) \\ k(\mathcal{S}_\ell, \mathbf{x}_{N+1})^T & 1 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} -K^{-1}k & \mathbf{I}_{N_\ell} \\ 1 & \mathbf{0}_{N_\ell}^T \end{bmatrix} \begin{bmatrix} \tau^{-1} & \mathbf{0}_{N_\ell}^T \\ \mathbf{0}_{N_\ell} & K(\mathcal{S}_\ell)^{-1} \end{bmatrix} \begin{bmatrix} -(K^{-1}k)^T & 1 \\ \mathbf{I}_{N_\ell} & \mathbf{0}_{N_\ell} \end{bmatrix} \end{aligned} \tag{10}$$

where $K^{-1}k = K(\mathcal{S}_\ell)^{-1} \cdot k(\mathcal{S}_\ell, \mathbf{x}_{N+1})$, $\mathbf{I}_{N_\ell}$ is the identity matrix of size N_ℓ and $\mathbf{0}_{N_\ell}$ is a vector of N_ℓ zeros. This formulation involves matrix-vector products of already computed terms instead of inverting a complete matrix (complexity $O(n^3)$). The computation works as long as $\tau(\boldsymbol{x}_{n+1}, \mathcal{M}_\ell)$ is different from 0, that is to say that the new point $\boldsymbol{x}_{n+1}$ should not belong to $\mathcal{S}_\ell$. In that case, we drop the point as its inclusion does not change the geometry of the manifold.

In its original form, PerTurbo algorithm requires as a learning stage to evaluate the Gram matrix corresponding to each learning set $\mathcal{S}_\ell$ and then to invert it, which leads to an $O(n(n-1)/2 + n^3)$ complexity. This operation has to be conducted for every class. Then at run time, computing the perturbation measure requires the evaluation of the kernel function for every test sample, and then conduct as many matrix-vector products as there are classes. While we can both note that: *i)* the number of learning samples N_ℓ for each class is significantly lower than the total number of learning samples N; *ii)* the method has particular advantages when the number of learning samples is low, it is still possible to develop strategies in order to alleviate the complexity burden. First, it is interesting to note that, due to the particular shape of the Gaussian kernel, the perturbation measure is a *local* measure, in the sense that distant samples will only bring small modifications to the perturbation measure. This leads to a fast yet accurate version of the algorithm, where only local perturbations are computed instead of global ones. In this new setting, there is no learning phase: no global Gram matrix is evaluated nor inversed. At testing phase, the learning samples can be sorted with respect to their distances to each test point. Then only a subset of t elements for each class can be used to form a local Gram matrix which is inverted and then used to measure the perturbation. In Figure 1, it is interesting to see that the perturbation measure is stabilising for small number t. How to set this parameter is crucially dependent of the kernel γ parameter. For a given γ, this parameter can be set empirically, with speed issues in mind (to the detriment of classification accuracy), or set by cross-validation.

This strategy has two consequences: since no learning phase is needed, the procedure can be really efficient for on-line or active learning problems (since no model of the class has to be updated when new samples arrive). This is a major advantage over SVM, which are not adapted to such problem configurations. Also, the complexity of the testing phase can be slightly raised since a sorting operation has to be conducted. This sorting operation dominates the complexity of the test, and can be efficiently carried in $O(n \log(n))$ operations, but we note that multidimensional spatial grid structures can be used to accelerate it in a classical fashion.

III. Experimental results on low-sized training sets

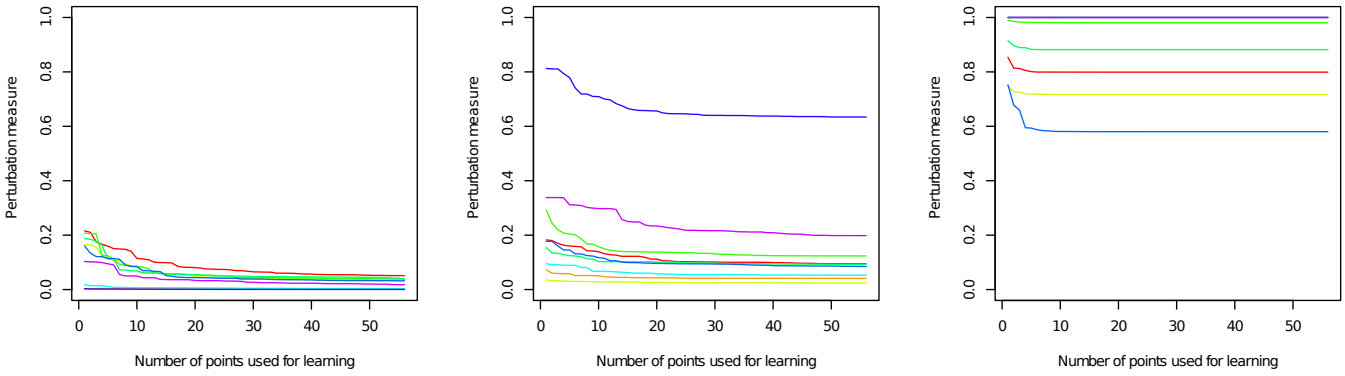
SVM are state-of-the-art techniques to perform hyperspectral image classification, and they have been shown to outperform other kernel techniques, even generative ones like Kernel fisher discriminant analysis, with low-sized training sets [22]. Hence, we focus only on SVM in our experiments. In order to have a fair comparison between PerTurbo and SVM, four standard hyperspectral scenes were used to carry out the experiments with different contexts (three urban areas and one agricultural scene).

A. Hyperspectral datasets

The first hyperspectral image used in our experiments is the *Indian Pines* scene, gathered by AVIRIS sensor over the agricultural Indian Pines test site in north-western Indiana. It represents a vegetation classification scenario, with two main crops of soybean and corn which are very early in their growth cycle. Indeed, the spectral information comes from crop mixtures, soil variation and crop residues remaining. Data are available through Purdue's university MultiSpec site[1], which made this scene widely benchmarked over

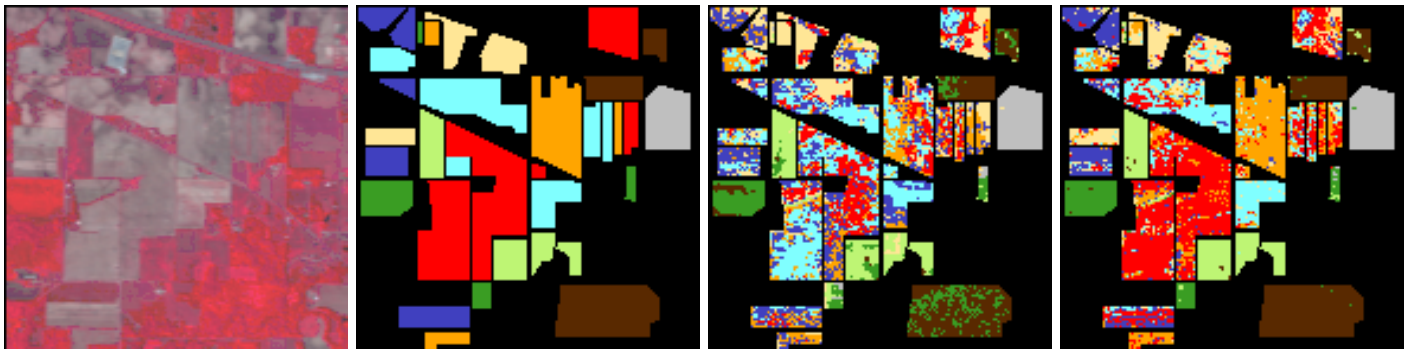
[1]https://engineering.purdue.edu/~biehl/MultiSpec/hyperspectral.html

Fig. 1. Evolution of the perturbation measure $\tau(\boldsymbol{\eta}, \mathcal{M}_\ell)$ for a test point $\boldsymbol{\eta}$ chosen randomly in the *Indian Pines* dataset for different t values and for 3 values of γ. Each line of each graph correspond to one class u_ℓ.



(a) $\gamma = 2^{-8}$ (b) $\gamma = 2^{-4}$ (c) $\gamma = 2$

Fig. 2. *Indian Pines* dataset. The colour code is the following: Corn-notill, Corn-mintill, Grass-pasture, Grass-trees, Hay-windrowed, Soybean-notill, Soybean-mintill, Soybean-clean, Woods.



(a) false colour composition (b) available reference data (c) PerTurbo classification results (5 points by class for training) (d) PerTurbo classification results (56 points by class for training)

the last few years. It consists of a 145×145 pixels image, in which $10,249$ pixels are labelled, and contains 200 spectral bands after discarding water absorption bands. From the initial sixteen classes of interest, we discard seven of them because their low number of pixels does not allow designing complete and fair experiments with a class-balanced set up (test set sizes should be great enough to allow randomisation). The reference map finally contains $9,234$ pixels. It is well-known that this dataset contains troublesome classes that are often misrecognised because of mislabelled and unclearly defined classes (see for instance [22]). For illustrative purpose, Figure 2 shows (a) a false colour composition of the scene and (b) the associated ground truth.

We also considered hyperspectral scenes of two urban areas, the *University of Pavia*, a 610×340 pixels image, and the *Center of Pavia*, a 1096×715 pixels image, both acquired by the ROSIS sensor. After removing some channels due to noise, the first scene contains 102 spectral bands and the second 103. Both images contain 9 reference classes that comprise urban, soil and vegetation features. *Pavia University* and *Center of Pavia* datasets contain respectively $46,697$ and $148,166$ labelled pixels.

The last dataset we used is the HYDICE image of *Washington DC mall* that comes along with the MultiSpec site. It is a 1280×307 pixels image, described initially by 210 spectral bands, whose opaque ones have been removed, leading to 191 bands. In order to remove noise on the data, we clipped 2% of the two-sided extreme values. $8,079$ pixels have been labelled among the 7 following references classes: roof, street, path, grass, tree, water and shadow.

B. Experimental setup

Each band of each original dataset has been scaled to the range $[0, 1]$. In all experiments, we restrict ourselves to Gaussian radial basis (RBF) kernels and parameter γ, that controls the spread of the kernel, has been set by trying exponentially growing sequences $\gamma = \left\{2^{-15}, 2^{-14}, ..., 2^2, 2^3\right\}$. We make here the assumption that classes have similar manifold geometry: it allows us to set a parameter γ constant between classes (we then choose a class-balanced set up in the experiments). Note that an unbalanced class scenario could also be considered, but at the price of setting an expensive grid search over γ_ℓ parameters (one distinct parameter per class, that would lead to p^L different values to test, where p is the number of γ_ℓ values and L the total number of classes). We made some preliminary experiments (not reported here) and the little improvement on the classification accuracies is not worth the exponential explosion of the cross-validation complexity.

The regularisation parameter λ needs to be set for PerTurbo classification method. In the case of the Tikhonov regularisation, we set $\lambda = \left\{10e^{-6}, 5e^{-5}, \cdots, 0\right\}$ or $\lambda = \{1, 0.999, 0.995, 0.99, 0.975, 0.95, 0.9, 0.75, 0.5\}$ in the truncated spectrum version of the algorithm. For SVM, the regularisation

parameter C, that controls the effect of errors on the classifier in order to regularise the separating surface, takes values $\{2^{-5}, 2^{-4}, ..., 2^{14}, 2^{15}\}$ with exponential steps. We use the *one-against-one* approach (see for instance [6]), in which $k(k-1)/2$ binary classifiers are trained; the appropriate class is then determined thanks to a voting scheme. We use R software for all our experiments: Kernlab implementation [33] to test SVM and pRoloc[2] for PerTurbo.

In order to compare the results of PerTurbo and SVM, we run a series of tests by varying the training set size from approximately 5 to 250 points per class (we choose a sequence of rounded numbers that roughly doubles the training set size at each increment), the maximimum values depending on the considered dataset: we stop increasing the size whether i) SVM significantly outperforms PerTurbo ii) the corresponding size does not allow the design of a fair and complete experiment. The size is defined as a proportion of the initial labelled set of the image, and the training set is formed as a randomly chosen class-balanced subset.

For each set of parameters, we compute two statistics:

- the overall accuracy (OA), which is the proportion of correctly classified pixels in the test set. Significance of the differences between the overall accuracies of SVM and the best regularised version of PerTurbo has been tested using the McNemar test [34]:
$$z_{OA} = \frac{f_{p-svm} - f_{svm-p}}{\sqrt{f_{p-svm} + f_{svm-p}}} \tag{11}$$
in which f_{p-svm} indicates the number of pixels correctly classified by PerTurbo and wrongly classified by SVM. A positive value of z_{OA} indicates that PerTurbo is more accurate than SVM. The difference between the overall accuracies of the two classifiers is statistically significant at 5% if $|z_{OA}| > 1.96$, at 1% if $|z_{OA}| > 2.58$ and at 0.1% if $|z_{OA}| > 3.29$.
- the kappa coefficient κ, which measures the agreement between predicted and actual class labels in the test set, corrected by agreement that could be expected by chance. Significance of the differences between the SVM kappa coefficient κ_{svm} and the best version of PerTurbo κ_p are tested using [34]:
$$z_\kappa = \frac{\kappa_p - \kappa_{svm}}{\sqrt{\sigma^2_{\kappa_p} + \sigma^2_{\kappa_{svm}}}} \tag{12}$$
where $\sigma^2_{\kappa_p}$ and $\sigma^2_{\kappa_{svm}}$ are the variance of the coefficients. The difference between the two coefficients is said significant at a 5% level if $|z_\kappa| > 1.96$.

Parameter γ depends on the geometry of the dataset and λ is related to the noise in the data: they are thus expected to differ between settings and datasets (note that a rule of thumb for γ has been given in [26] and can be used as a first try). We then use grid search methods to assess the generalisation ability of the algorithm. As the setting we consider is a low number of training samples, we cannot use classical cross-validation schemes to set the parameters. Instead, we use a repeated random sub-sampling scheme, by running 50 experiments on each couple (γ, λ) or (γ, C), and we report the results for parameters that lead to the best overall accuracies.

C. Results

We compare the two methods regarding their ability to deal with a low number of training samples, which is a crucial problem in hyperspectral image classification as the labelling cost can be important. We then particularly focus on ill-posed problems, where the input dimension is higher than the number of training samples. Table I gives, for different small training set sizes, the best overall accuracies, the associated kappa coefficient and z_{OA} values of the McNemar test. Performances are given for SVM, the truncated spectrum version of PerTurbo and PerTurbo with Tikhonov regularisation. We first note that PerTurbo with the Tikhonov regularisation outperforms the truncated spectrum version of the algorithm, for all experiments but one. Following conclusions are then done by considering only the Tikhonov regularisation version of PerTurbo. For illustration purpose, Table II gives the computational times obtained for the reported experiments: training phase of PerTurbo is from 2 to 7 times faster than SVM while PerTurbo's testing time is lower than SVM for the lowest training set sizes but SVM have advantage when the training set size increases. This is mainly due to the fact that SVM only require the computation of the similarities between the test data and the support vectors, whereas PerTurbo testing step involves all the similarities if the policy described in Section II-C is not applied.

We observe that PerTurbo yields the best overall accuracies for the lowest training set sizes. For *Indian Pines* dataset, conclusions have to be mitigated: only the lowest training set size gives advantage to PerTurbo over SVM. Figure 2 illustrates the classification results obtained with (c) the lowest and (d) the highest training set sizes that have been considered. We also run additional experiments for the lowest training set sizes that include all the classes (not reported here for the sake of concision), and the same conclusions can be raised, except that PerTurbo significantly outperforms SVM for the lowest training set size. In our opinion, the results are mainly due to the high proportion of mislabelled data among the few training samples available: the ground truth map labels entire fields with a single class reference, even though the ground cover of corn or soybean varies inside. We then assess the separability of the manifold of each class following Eq. (9). Calculating the similarity matrix between each class and its surrogates gives a table layout; a function of the obtained values, as well as a function of the intensity of the values of the confusion matrix, are represented in Figure 3. One can notice the structural similarity of the two matrices, confirming the idea that an a priori on the classification results can be obtained from the computation of the similarity matrix. More specifically, we note that classes representing corn and soybean are badly separated by PerTurbo as they have similar geometric characterisation. On the contrary, class *Grass-Trees* is well predicted as its manifold lies in different space than those of the other classes.

For *Pavia* datasets, classification accuracies between PerTurbo and SVM are different at a 5% level of significance: PerTurbo outperforms SVM when the training set sizes are the lowest (less or equal than $N_\ell = 52$ per class for *Pavia University*, $N_\ell = 41$ per class for *Pavia Center*). Regarding the kappa coefficient, the same conclusion can be drawn, except that the differences between coefficients are hardly significant. As the rate of training samples increases, SVM take advantage over PerTurbo, both in terms of overall accuracy and kappa coefficient.

Regarding the *Washington DC mall* dataset, experiments show again that PerTurbo gives significant higher accuracy rates when the training set size is low. In this particular case and with the tested training set sizes, we do not observe the expected higher asymptotic accuracy rate of SVM over PerTurbo.

IV. Conclusion

As indicated in a recent survey [1], hyperspectral data classification offers new perspectives to remote sensing but also raises fundamental challenges: (i) unbalance between high dimensionality and limited availability of training samples, (ii) complex geometry of hyperspectral data dominated by non-linear structures, (iii) very high computational complexity of classification techniques in presence of hyperspectral data of very large dimensionality and complexity.

In this paper, we have addressed all these challenges by focusing on generative techniques, and especially those representing hyperspectral data as manifolds. To do so, we have considered PerTurbo, a recent representative algorithm of the field, and benchmarked it against traditional SVM. The results observed on standard datasets are appealing and demonstrate the interest of such tools when the number of samples is very low, which is often the case with hyperspectral images. Indeed, PerTurbo exhibits better performances on smallest training set sizes, in comparison to SVM that are expected to have an

[2]http://www.bioconductor.org/packages/2.12/bioc/html/pRoloc.html

TABLE I

MEAN ± STANDARD DEVIATION OF OVERALL ACCURACIES (OA), KAPPA (κ) STATISTICS AND MC NEMAR TEST VALUES z_{OA} COMPUTED OVER 50 REPETITIONS FOR *Indian Pines*, *Pavia University*, *Pavia Center* AND *Washington DC mall* DATASETS. TRAINING SET SIZE % GIVES THE PROPORTION OF THE DATASET THAT IS USED FOR TRAINING WHILE # GIVES THE NUMBER OF POINTS PER CLASS THAT COMPOSE THE TRAINING SET. FOR EACH TRAINING SET SIZE, BEST RESULTS ARE REPORTED IN BOLD FACE. RESULTS BEING STATISTICALLY BETTER (BETWEEN SVM AND THE BEST VERSION OF PERTURBO) AT A 5% LEVEL ARE UNDERLINED. z_{OA} VALUES ARE CALCULATED CONSIDERING SVM AND THE THIKHONOV VERSION OF PERTURBO: POSITIVE VALUES INDICATE THAT PERTURBO OUTPERFORMS SVM.

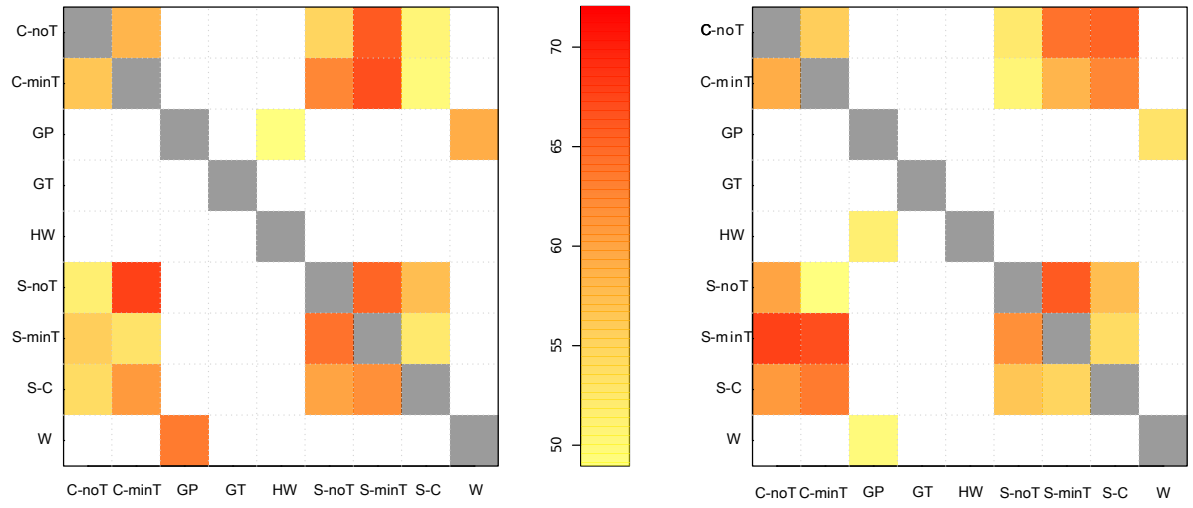
INDIAN PINES							
Training set size	**SVM**		**PerTurbo (truncated)**		**PerTurbo (Tikhonov)**		z_{OA}
(% / #)	**OA[%]**	**κ[%]**	**OA[%]**	**κ[%]**	**OA[%]**	**κ[%]**	
0.5 / 5	51.6 ± 3.0	44.6 ± 3.2	51.7 ± 3.1	44.7 ± 3.2	**51.8** ± 3.1	**44.8** ± 3.2	0.5
1 / 11	**59.1** ± 2.9	**53.0** ± 3.2	58.4 ± 2.9	52.1 ± 3.2	58.5 ± 2.5	52.3 ± 3.2	-1.7
2.5 / 26	**68.6** ± 1.8	**63.7** ± 2.0	65.7 ± 1.9	60.3 ± 2.1	66.1 ± 1.6	60.7 ± 2.0	-6.1
5 / 56	**75.2** ± 1.1	**71.2** ± 1.3	70.8 ± 1.0	66.0 ± 1.1	71.3 ± 1.1	66.5 ± 1.2	-8.9
PAVIA UNIVERSITY							
Training set size	**SVM**		**PerTurbo (truncated)**		**PerTurbo (Tikhonov)**		z_{OA}
(% / #)	**OA[%]**	**κ[%]**	**OA[%]**	**κ[%]**	**OA[%]**	**κ[%]**	
0.1 / 5	61.7 ± 5.4	53.7 ± 5.5	**64.5** ± 5.4	**56.4** ± 5.6	63.8 ± 4.0	55.8 ± 4.1	3.7
0.25 / 13	74.4 ± 3.5	68.1 ± 3.9	74.4 ± 2.8	68.2 ± 3.4	**75.6** ± 3.2	**69.6** ± 3.5	5.8
0.5 / 26	79.9 ± 2.1	74.7 ± 2.4	78.2 ± 1.7	72.7 ± 1.9	**81.8** ± 1.8	**77.0** ± 2.0	9.6
1 / 26	83.8 ± 1.7	79.7 ± 1.9	81.9 ± 77.2	77.2 ± 1.7	**85.1** ± 1.5	**81.2** ± 1.7	6.6
2.5 / 130	**89.6** ± 0.7	**86.6** ± 0.9	84.7 ± 1.0	80.5 ± 1.2	86.9 ± 0.8	83.3 ± 0.9	-15.8
5 / 259	**91.4** ± 0.5	**88.8** ± 0.6	86.7 ± 0.8	82.8 ± 0.9	88.2 ± 0.9	84.9 ± 1.0	-19.7
PAVIA CENTER							
Training set size	**SVM**		**PerTurbo (truncated)**		**PerTurbo (Tikhonov)**		z_{OA}
(% / #)	**OA[%]**	**κ[%]**	**OA[%]**	**κ[%]**	**OA[%]**	**κ[%]**	
0.025 / 4	90.6 ± 2.8	86.8 ± 3.9	90.5 ± 2.8	86.7 ± 3.8	**92.1** ± 1.2	**89.0** ± 1.6	23.3
0.05 / 8	93.6 ± 1.2	91.1 ± 2.8	93.6 ± 1.1	90.8 ± 1.6	**93.9** ± 1.1	**91.6** ± 1.6	4.3
0.1 / 16	95.5 ± 0.5	93.7 ± 0.7	95.6 ± 0.6	93.8 ± 0.8	**95.7** ± 0.6	**93.9**±0.8	4.4
0.25 / 41	96.6 ± 0.4	95.2 ± 0.5	96.7 ± 0.3	95.5 ± 0.4	**97.0** ± 0.3	**95.8** ± 0.6	11.3
0.5 / 82	**97.5** ± 0.3	**96.8** ± 0.4	96.8 ± 0.3	95.5 ± 0.4	97.1 ± 0.2	95.9 ± 0.3	-11.9
1 / 165	**98.2** ± 0.2	**97.4** ± 0.3	97.0 ± 0.4	95.7 ± 0.6	97.1 ± 0.1	95.9 ± 0.2	-28.6
WASHINGTON DC MALL							
Training set size	**SVM**		**PerTurbo (truncated)**		**PerTurbo (Tikhonov)**		z_{OA}
(% / #)	**OA[%]**	**κ[%]**	**OA[%]**	**κ[%]**	**OA[%]**	**κ[%]**	
0.5 / 6	94.7 ± 4.7	92.3 ± 6.2	93.2 ± 5.9	90.5 ± 7.7	**94.9** ± 5.9	**93.0** ± 7.8	3.4
1 / 12	98.0 ± 1.3	97.1 ± 2.5	98.2 ± 1.9	97.4 ± 2.6	**98.7** ± 0.8	**98.2** ± 1.2	4.0
2.5 / 29	99.1 ± 0.4	98.6 ± 0.6	97.8 ± 1.1	96.8 ± 1.6	**99.2** ± 0.4	**98.9** ± 0.5	2.1
5 / 58	99.4 ± 0.3	99.1 ± 0.4	99.6 ± 0.3	99.4 ± 0.5	**99.7** ± 0.2	**99.5** ± 0.3	3.3

TABLE II

COMPUTATION TIMES (IN SECONDS, AVERAGED OVER THE 50 REPETITIONS) FOR SVM AND PERTURBO TRAINING AND TESTING PHASES. THE TWO VERSIONS OF PERTURBO EXHIBIT A SLIGHT DIFFERENCE IN THE REPORTED EXPERIMENTS; RESULTS FOR THE TIKHONOV VERSION ARE ONLY REPORTED HERE. LOWEST COMPUTATIONAL TIMES ARE REPORTED BOLDFACED.

INDIAN PINES					**WASHINGTON DC MALL**				
Training set size	**SVM**		**PerTurbo**		**Training set size**	**SVM**		**PerTurbo**	
(% / #)	**Train**	**Test**	**Train**	**Test**	**(% / #)**	**Train**	**Test**	**Train**	**Test**
0.5 / 5	0.09	1.07	**0.02**	**0.63**	0.5 / 6	0.06	0.49	**0.01**	**0.35**
1 / 11	0.14	1.16	**0.02**	**0.82**	1 / 12	0.08	0.57	**0.01**	**0.38**
2.5 / 26	0.19	1.44	**0.02**	**0.96**	2.5 / 29	0.13	0.56	**0.02**	**0.49**
5 / 56	0.28	1.33	**0.03**	**1.29**	5 / 58	0.20	0.77	**0.02**	**0.69**
PAVIA UNIVERSITY					**PAVIA CENTER**				
Training set size	**SVM**		**PerTurbo**		**Training set size**	**SVM**		**PerTurbo**	
(% / #)	**Train**	**Test**	**Train**	**Test**	**(% / #)**	**Train**	**Test**	**Train**	**Test**
0.1 / 5	0.08	2.44	**0.02**	**2.23**	0.025 / 4	0.08	8.21	**0.02**	**7.52**
0.25 / 13	0.09	**2.61**	**0.02**	2.73	0.05 / 8	0.08	8.88	**0.02**	**7.99**
0.5 / 26	0.11	**2.90**	**0.02**	3.16	0.1 / 16	0.09	9.46	**0.02**	**8.82**
1 / 52	0.17	**3.28**	**0.03**	4.84	0.25 / 41	0.17	**9.90**	**0.03**	16.17
2.5 / 130	0.41	**4.20**	**0.10**	11.94	0.5 / 82	0.23	**9.64**	**0.06**	24.93
5 / 259	0.84	**5.39**	**0.44**	28.03	1 / 165	0.43	**11.63**	**0.16**	55.78

Fig. 3. Measures of separability of the different classes of the *Indian Pines* dataset. Cells represent, in Figure (a) a function of the measure of similarity $A(\mathcal{S}_{\ell_1}, K(\mathcal{S}_{\ell_1} \to \mathcal{M}_{\ell_2}), K(\mathcal{S}_{\ell_1}))$ between training set $\mathcal{S}_{\ell_1}$ of the class in column u_{ℓ_1} composed of $N_{\ell_1} = 26$ pixels, projected in the manifold of the class in row u_{ℓ_2}; in Figure (c), a function of the percentage of pixels of class in column that have been classified as the class in row. The function used is the rank; colour intensity in Figure b) represents the rank of each cell: the most similar pairs of classes and those with highest error rates being coloured in dark red (cells with the lowest ranks are not coloured and diagonal cells are coloured in grey in both cases). The class code is the following: C-noT: Corn-notill, C-minT: Corn-mintill, GP: Grass-pasture, GT: Grass-trees, HW: Hay-windrowed, S-noT: Soybean-notill, S-minT: Soybean-mintill, S-C: Soybean-clean, W: Woods.



(a) Similarity matrix (b) Ranks (c) Confusion matrix

higher asymptotic accuracy rate. In addition, the geometric characterisation of each class allows one to derive a similarity measure between each pair of classes that can then be interpreted by the practitioner, *e.g.* give beforehand an insight into the class separability.

Among the other challenges which remain to be addressed [1], the combination of spatial and spectral information as well as the attention paid to mixed pixels in the data have to be tackled. In addition, the required size of the training set such that SVM performs well (and then is expected to outperform PerTurbo) depends on the degree to which the training set contains pixels that lie at the edge of the class distribution in the feature space: a sparse feature space will thus require more training points to get an accurate discriminative decision surface. It will be interesting to derive a threshold in order to determine beforehand which method should be preferred in terms of performances for a given training set size. These are some of the future directions of our work.

V. Acknowledgments

The authors acknowledge the support of the French Agence Nationale de la Recherche (ANR) under reference ANR-13-JS02-0005-01 (Asterix project). They also wish to thank Paolo Gamba from University of Pavia and the HySens project for providing the data included in the Pavia datasets.

References

[1] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, "Hyperspectral remote sensing data analysis and future challenges," *IEEE Geoscience and Remote Sensing Magazine*, vol. 1, no. 2, pp. 6–36, 2013.

[2] D. A. Landgrebe, *Signal Theory Methods in Multispectral Remote Sensing*, Wiley, 2003.

[3] G.H. Hughes, "On the mean accuracy of statistical pattern recognizers," *IEEE Transactions on Information Theory*, vol. 14, pp. 55–63, 1968.

[4] K. Fukunaga and R.R. Hayes, "Effects of sample size in classifier design," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 873–885, 1989.

[5] V. N. Vapnik, *Statistical learning theory*, Wiley, 1998.

[6] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, pp. 1778–1790, 2004.

[7] B. Guo, S. Gunn, R. Damper, and J. Nelson, "Customizing kernel functions for SVM-based hyperspectral image classification," *IEEE Transactions on Image Processing*, vol. 44, pp. 2839–2846, 2008.

[8] M. Fauvel, J. Chanussot, and J.A. Benediktsson, "A spatial-spectral kernel-based approach for the classification of remote-sensing images," *Pattern Recognition*, vol. 45, pp. 381–392, 2012.

[9] B. Schölkopf, A. Smola, and K. R. Müller, "Kernel principal component analysis," in *Artificial Neural Networks (ICANN)*, vol. 1327 of *Lecture Notes in Computer science*, pp. 583–588. Springer, 1997.

[10] J. Lafferty and G. Lebanon, "Diffusion kernels on statistical manifolds," *Journal of Machine Learning Research*, vol. 6, pp. 129–163, 2005.

[11] A. Green, M. Berman, P. Switzer, and M. Craig, "A transformation for ordering multispectral data in terms of image quality with implications for noise removal," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 26, no. 1, pp. 65–74, 1988.

[12] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[13] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[14] B. Hou, X. Zhang, Q. Ye, and Y. Zheng, "A novel method for hyperspectral image classification based on laplacian eigenmap pixels distribution-flow," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 3, pp. 1602–1618, 2013.

[15] C. M. Bachmann, T. L. Ainsworth, and R. A. Fusina, "Exploiting manifold geometry in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 3, pp. 441–454, 2005.

[16] T. Han and D. G. Goodenough, "Nonlinear feature extraction of hyperspectral data based on locally linear embedding (lle)," in *IEEE International conference on Geoscience and Remote Sensing Symposium (IGARSS)*, 2005, vol. 2, pp. 1237–1240.

[17] W. Kim and M. Crawford, "Adaptive classification for hyperspectral image data using manifold regularization kernel machines," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 11, pp. 4110–4121, 2010.

[18] M. Li, M. M. Crawford, and J. Tian, "Local manifold learning-based k-nearest-neighbor for hyperspectral image classification," *IEEE*

Transactions on Geoscience and Remote Sensing, vol. 48, no. 11, pp. 4099–4109, 2010.
[19] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in *Advances in Neural Information Processing Systems (NIPS)*, 2001, pp. 841–848.
[20] T. Hospedales, S. Gong, and T. Xiang, "Finding rare classes: Active learning with generative and discriminative models," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 2, pp. 374–386, 2013.
[21] J. H. Xue and D. M. Titterington, "Comment on "on discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes"," *Neural Processing Letters*, vol. 28, no. 3, pp. 1370–4621, 2008.
[22] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, pp. 1351–1362, 2005.
[23] S. Wenzel and L. Hotz, "The role of sequences for incremental learning," in *International Conference on Agents and Artificial Intelligence (ICAART)*, 2010, vol. 1, pp. 434–439.
[24] N. Courty, T. Burger, and L. Johann, "PerTurbo: a new classification algorithm based on the spectrum perturbations of the laplace-beltrami operator," in *European Conference on Machine Learning and Practice and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, 2011, vol. 1, pp. 359–374.
[25] I. Chavel, *Eigenvalues in Riemannian geometry*, Academic Press, 1984.
[26] L. Chapel, T. Burger, N. Courty, and S. Lefèvre, "Classwise hyperspectral image classification with PerTurbo method," in *IEEE International conference on Geoscience and Remote Sensing Symposium (IGARSS)*, 2012, pp. 6883–6886.
[27] C. Öztireli, M. Alexa, and M. Gross, "Spectral sampling of manifolds," in *ACM Transaction on Graphics (Siggraph Asia)*, 2010.
[28] H. Cevikalp, D. Larlus, M. Neamtu, B. Triggs, and F. Jurie, "Manifold based local classifiers: Linear and nonlinear approaches," *Journal of Signal Processing Systems*, vol. 61, no. 1, pp. 61–73, 2010.
[29] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems (NIPS)*, 2001, vol. 14, pp. 585–591.
[30] K. Zhang, V. Zheng, Q. Wang, J. Kwok, Q. Yang, and I. Marsic, "Covariate shift in hilbert space: A solution via surrogate kernels," in *International Conference on Machine Learning (ICML)*, 2013, vol. 28, pp. 388–395.
[31] N. Shawe-Taylor and A. Kandola, "On kernel target alignment," in *Advances in Neural Information Processing Systems (NIPS)*, 2002, vol. 14, p. 367.
[32] S. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge University Press, 2004.
[33] A. Karatzoglou, A. Smola, K. Hornik, and A. Zeileis, "kernlab – An S4 package for kernel methods in R," *Journal of Statistical Software*, vol. 11, no. 9, pp. 1–20, 2004.
[34] G. M. Foody, "Thematic map comparison: evaluating the statistical significance of differences in classification accuracy," *Photogrammetric Engineering & Remote Sensing*, vol. 70, no. 5, pp. 627–633, 2004.

Laetitia Chapel received the B.Sc. degree in statistics and computer science from Université de Bretagne-Sud, France in 2002, the M.Sc. degree in knowledge discovery in databases in 2004 and the Ph.D. degree in computer science from Université Blaise Pascal in 2007. She was a post-doctoral research fellow with the Hamilton Institute, National University of Ireland Maynooth from 2008 to 2010. She is currently an associate professor in statistics and computer science in IRISA, Université de Bretagne-Sud. Her main research topic is machine learning in general and with applications in remote sensing.

Thomas Burger is a CNRS researcher with iRTSV, in a lab devoted to the exploration of the dynamics of proteomes. After two MS degrees (in computer sciences and in discrete mathematics) in 2004, he defended a PhD in image and video processing in 2007. Then, he joined Université de Bretagne-Sud as an associate professor in statistics and machine learning. In 2011, he joined his current position, where he became the Knowledge Discovery from Data group leader in 2013. His current fields of interest are machine learning, uncertainty representation, bioinformatics and data mining.

Nicolas Courty obtained in 1999 an engineer degree in computer science from INSA Rennes, and a Master degree from University of Rennes I (France). He obtained his Ph.D. degree from Insa Rennes in 2002 on the subject of image-based animation techniques. He then spent one year as post doctoral student in Porto Alegre, Brazil, with an INRIA fellowship. He integrated the Université de Bretagne-Sud (in Vannes, France) as assistant professor in 2004. In 2012, he spent 8 months in the Chinese Academy of Science Institute of Automation (CASIA), in Beijing, China, with a Visiting Professorships for Senior International Scientists grant. His current research interests include analysis/synthesis schemes for dynamical phenomena, inverse problems and machine learning for computer graphics/vision.

Sébastien Lefèvre received his M.Sc. and Eng. degrees in Computer Engineering from the University of Technology of Compiègne in 1999, and his Ph.D. in Computer Science from the University of Tours in 2002. In 2009, he earned his French HDR degree in Computer Science from the University of Strasbourg. From 2003 to 2010, he was an Associate Professor in the Department of Computer Sciences and the Image Sciences, Computer Sciences and Remote Sensing Laboratory (LSIIT), University of Strasbourg - CNRS. In 2009-2010, he was an INRIA invited scientist within the TEXMEX team of IRISA/INRIA Rennes. In 2010, he joined the Université de Bretagne-Sud as a Full Professor in Computer Science, in the Institute of Technology of Vannes and the Institute for research in computer science and random systems (IRISA). Within IRISA, he is leading the OBELIX team dedicated to image analysis and machine learning for remote sensing and earth observation. His research interests are in image analysis and pattern recognition, using mainly mathematical morphology, hierarchical models and machine learning techniques with applications in earth observation and content-based image retrieval.